What is claimed is:

1. A system for efficiently forwarding client requests in a distributed computing environment, comprising:

a socket receiving a plurality of non-proxiable requests commonly addressed to an origin server from individual sending clients;

a time estimates generator dynamically generating, concurrent to and during processing of each request, time estimates of service availability based on a time-to-idle for sending the requests over each of a plurality of connections to the origin server; and

a connection manager selecting the connection to the origin server with a substantially highest service availability and a substantially lowest time-to-idle and forwarding each request to the origin server using the selected connection.

2. A system according to Claim 1, further comprising:

the connection manager selecting a connection not actively sending a request with a zero time-to-idle and not subject to a slow start overhead incurred responsive to flow control imposed by the sending client.

3. A system according to Claim 2, further comprising:

the connection manager selecting a connection actively sending a request with a time-to-idle less than the slow start overhead, plus request transfer time if the connection is pipelined.

4. A system according to Claim 3, further comprising:

the connection manager selecting a connection not actively sending a request with a zero time-to-idle and subject to the slow start overhead.

5. A system according to Claim 4, further comprising:

the connection manager selecting a connection actively sending a request with a time-to-idle less than a connection setup overhead, plus request transfer time if the connection is pipelined.

6. A system according to Claim 5, further comprising:

the connection manager selecting a new connection in the absence of an existing connection with a time-to-idle less than the connection setup overhead.

7. A system according to Claim 5, further comprising:

the connection manager selecting an existing connection with the substantially lowest time-to-idle.

8. A system according to Claim 1, wherein the distributed operating environment is TCP/IP-compliant, the system further comprising:

the time estimates generator providing time estimates for each connection comprising at least one of TCP overhead, time-to-idle, idle time, and request transfer time.

9. A system according to Claim 8, the connection setup overhead comprises TCP overhead, the system further comprising:

the time estimates generator calculating the TCP overhead by adding a three-way handshake overhead to a slow start overhead.

10. A system according to Claim 8, further comprising:

the time estimates generator calculating the request transfer time by multiplying the size of the request by an average connection speed for the origin server.

11. A system according to Claim 8, further comprising:

the time estimates generator calculating the time-to-idle upon each receipt of a request by adding the time-to-idle to the product of an average connection speed for the origin server multiplied by the sum of the request size and an estimated response size.

12. A system according to Claim 8, further comprising:

the time estimates generator calculating the time-to-idle upon writing data to a socket by subtracting the time-to-idle from the product of an average connection speed for the origin server multiplied by the amount of data written.

13. A system according to Claim 8, further comprising:

the time estimates generator calculating the time-to-idle upon reading data from a socket, prior to header data, by subtracting the time-to-idle from the product of an average connection speed for the origin server multiplied by the amount of data read.

14. A system according to Claim 1, further comprising:

a proxy configured in a location comprising at least one of local to the sending clients, in the infrastructure of the distributed computing environment, and local to the origin server.

15. A method for efficiently forwarding client requests in a distributed computing environment, comprising:

receiving a plurality of non-proxiable requests commonly addressed to an origin server from individual sending clients;

dynamically generating, concurrent to and during processing of each request, time estimates of service availability based on a time-to-idle for sending the requests over each of a plurality of connections to the origin server; and

selecting the connection to the origin server with a substantially highest service availability and a substantially lowest time-to-idle and forwarding each request to the origin server using the selected connection.

16. A method according to Claim 15, further comprising:

selecting a connection not actively sending a request with a zero time-to-idle and not subject to a slow start overhead incurred responsive to flow control imposed by the sending client.

17. A method according to Claim 16, further comprising:

selecting a connection actively sending a request with a time-to-idle less than the slow start overhead, plus request transfer time if the connection is pipelined.

18. A method according to Claim 17, further comprising:

selecting a connection not actively sending a request with a zero time-to-idle and subject to the slow start overhead.

19. A method according to Claim 18, further comprising:

selecting a connection actively sending a request with a time-to-idle less than a connection setup overhead, plus request transfer time if the connection is pipelined.

20. A method according to Claim 19, further comprising:

selecting a new connection in the absence of an existing connection with a time-to-idle less than the connection setup overhead.

21. A method according to Claim 19, further comprising:

selecting an existing connection with the substantially lowest time-to-idle.

22. A method according to Claim 15, wherein the distributed operating environment is TCP/IP-compliant, the method further comprising:

providing time estimates for each connection comprising at least one of TCP overhead, time-to-idle, idle time, and request transfer time.

23. A method according to Claim 22, the connection setup overhead comprises TCP overhead, the method further comprising:

calculating the TCP overhead by adding a three-way handshake overhead to a slow start overhead.

24. A method according to Claim 22, further comprising:

calculating the request transfer time by multiplying the size of the request by an average connection speed for the origin server.

25. A method according to Claim 22, further comprising:

calculating the time-to-idle upon each receipt of a request by adding the time-to-idle to the product of an average connection speed for the origin server multiplied by the sum of the request size and an estimated response size.

26. A method according to Claim 22, further comprising:

calculating the time-to-idle upon writing data to a socket by subtracting the time-to-idle from the product of an average connection speed for the origin server multiplied by the amount of data written.

27. A method according to Claim 22, further comprising:

calculating the time-to-idle upon reading data from a socket, prior to header data, by subtracting the time-to-idle from the product of an average connection speed for the origin server multiplied by the amount of data read.

28. A method according to Claim 15, further comprising:

providing a proxy configured in a location comprising at least one of local to the sending clients, in the infrastructure of the distributed computing environment, and local to the origin server.

29. A computer-readable storage medium holding code for performing the method according to Claim 15.

30. A system for efficiently forwarding client requests from a proxy server in a TCP/IP computing environment, comprising:

means for receiving a plurality of transient requests from individual sending clients, each request being commonly addressed to an origin server;

means for dynamically calculating, concurrent to receiving and during processing of each request, time estimates of TCP overhead, slow start overhead, time-to-idle, and request transfer time for sending the requests over each of a plurality of managed connections to the origin server;

means for choosing the managed connection from, in order of preferred selection, a warm idle connection, an active connection with a time-to-idle less than a slow start overhead, a cold idle connection, an active connection with a time-to-idle less than a TCP overhead, a new managed connection, and an existing managed connection with a smallest time-to-idle; and

means for forwarding each request to the origin server over the selected managed connection.

31. A system according to Claim 30, further comprising:

means for adding the request transfer time during each active connection selection if the managed connection is pipelined.

32. A system according to Claim 30, further comprising:

means for calculating the TCP overhead by adding a three-way handshake overhead to a slow start overhead;

means for calculating the request transfer time by multiplying the size of the request by an average managed connection speed for the origin server; and

means for calculating the time-to-idle, comprising:

upon each receipt of a request, means for adding the time-to-idle to the product of an average managed connection speed for the origin server multiplied by the sum of the request size and an estimated response size;

upon writing data to a socket, means for subtracting the time-to-idle from the product of an average managed connection speed for the origin server multiplied by the amount of data written; and

upon reading data from a socket, prior to header data, means for subtracting the time-to-idle from the product of an average managed connection speed for the origin server multiplied by the amount of data read.

33. A system according to Claim 30, wherein each transient request is communicated in accordance with HTTP.

34. A method for efficiently forwarding client requests from a proxy server in a TCP/IP computing environment, comprising:

receiving a plurality of transient requests from individual sending clients into a request queue, each request being commonly addressed to an origin server;

dynamically calculating, concurrent to receiving and during processing of each request, time estimates of TCP overhead, slow start overhead, time-to-idle, and request transfer time for sending the requests over each of a plurality of managed connections to the origin server;

choosing the managed connection from, in order of preferred selection, a warm idle connection, an active connection with a time-to-idle less than a slow start overhead, a cold idle connection, an active connection with a time-to-idle less than a TCP overhead, a new managed connection, and an existing managed connection with a smallest time-to-idle; and

forwarding each request to the origin server over the selected managed connection.

35. A method according to Claim 34, further comprising:

adding the request transfer time during each active connection selection if the managed connection is pipelined.

36. A method according to Claim 34, further comprising:

calculating the TCP overhead by adding a three-way handshake overhead to a slow start overhead;

calculating the request transfer time by multiplying the size of the request by an average managed connection speed for the origin server; and

calculating the time-to-idle, comprising:

upon each receipt of a request, adding the time-to-idle to the product of an average managed connection speed for the origin server multiplied by the sum of the request size and an estimated response size;

upon writing data to a socket, subtracting the time-to-idle from the product of an average managed connection speed for the origin server multiplied by the amount of data written; and

upon reading data from a socket, prior to header data, subtracting the time-to-idle from the product of an average managed connection speed for the origin server multiplied by the amount of data read.

37. A method according to Claim 34, wherein each transient request is communicated in accordance with HTTP.

38. A computer-readable storage medium holding code for performing the method according to Claim 34.